Machine Learning & Prediction

Irene Yao

5/2/2018

Based on the previous data analysis (https://github.com/ireneyaoyao/Springboard/blob/master/Capstone/Statistical%20Analysis.pdf), we will use the following variables for prediction of outcome type.

- size
- intake_condition
- outcome_condition
- sex
- age_at_intake
- stage_at_outcome (age)
- days in shelter
- breed (is mix or not)

Machine Learning and Prediction

For this report's purpose, the prediction will be made around dogs, and the predicted value will be the outcome of each animal. I will use both GBM and XGBoost for the prediction and the outcome will be a binary classification. The outcomes for the animals will be either "placed in a home" or "not placed in a home". "Adoption" and "Return to owner" will regarded as "placed in a home", and all others will be categorized into "not place in a home". To do so, I will add a column "placed" and the binary value for the column will be 1 or 0.

```
dogs$placed <- ifelse((dogs$outcome_type == "ADOPTION" | dogs$outcome_type == "RETURN TO OWNER"
), 1, 0)
```

Some of the columns have a class of "character" or "timediff". In order for the prediction model to work, update those columns to either "factors" or "numeric".

```
dogs$sex_clean <- as.factor(dogs$sex_clean)
dogs$stage_at_outcome <- as.factor(dogs$stage_at_outcome)
dogs$age_at_intake <- as.numeric(dogs$age_at_intake)
dogs$age_at_outcome <- as.numeric(dogs$age_at_outcome)
```

Prediction Using GBM

1. separate the dataframe into a training and a testing set. 80% of data will be in training set and the rest 20% will be in testing set.

```
n <- nrow(dogs)
n_train <- round(n * 0.8)
set.seed(123)
train_indices <- sample(1:n, n_train)
dog_train <- dogs[train_indices, ]
dog_test <- dogs[-train_indices, ]
```

2. create the GBM model

```
library(gbm)
set.seed(1)
dog_model_gbm <- gbm(formula = placed ~ size + intake_condition + outcome_condition + sex_clean
 + age_at_intake + stage_at_outcome + is_mix,
                             distribution = "bernoulli",
                             data = dog_train,
                             n.trees = 10000)
```

3. predict the outcomes of the test set

```
pred_gbm <- predict(object = dog_model_gbm,
                newdata = dog_test,
                n.trees = 10000,
                type = "response")
```

4. evaluate the model using test set AUC

```
library(Metrics)
auc_gbm <- auc(actual=dog_test$placed, predicted=pred_gbm)
print(paste0("Test set AUC: ", auc_gbm))
```

```
## [1] "Test set AUC: 0.873844537815126"
```

5. evaluate the model using RMSE

```
rmse_gbm <- dog_test %>%
  mutate(residuals = placed - pred_gbm) %>%
  summarize(rmse = sqrt(mean(residuals^2)))
```

Prediction Using XGBoost

1. prepare both the training and testing data using vtreat

```
library(vtreat)

vars <- c("size","intake_condition","outcome_condition","sex_clean","age_at_intake","stage_at_ou
tcome","is_mix")

treatplan <- designTreatmentsZ(dog_train, vars)
```

```
## [1] "vtreat 1.0.3 inspecting inputs Wed May 02 20:55:34 2018"
## [1] "designing treatments Wed May 02 20:55:34 2018"
## [1] " have initial level statistics Wed May 02 20:55:34 2018"
## [1] "design var size Wed May 02 20:55:34 2018"
## [1] "design var intake_condition Wed May 02 20:55:34 2018"
## [1] "design var outcome_condition Wed May 02 20:55:35 2018"
## [1] "design var sex_clean Wed May 02 20:55:35 2018"
## [1] "design var age_at_intake Wed May 02 20:55:35 2018"
## [1] "design var stage_at_outcome Wed May 02 20:55:35 2018"
## [1] "design var is_mix Wed May 02 20:55:35 2018"
## [1] " scoring treatments Wed May 02 20:55:35 2018"
## [1] "have treatment plan Wed May 02 20:55:35 2018"
```

```
scoreFrame <- treatplan %>%
    magrittr::use_series(scoreFrame) %>%
    select(varName, origName, code)

newvars <- scoreFrame %>%
  filter(code %in% c("clean", "lev")) %>%
  magrittr::use_series(varName)

dog_train_treat <- prepare(treatplan, dog_train, varRestriction = newvars)

treatplan_test <- designTreatmentsZ(dog_test, vars)
```

```
## [1] "vtreat 1.0.3 inspecting inputs Wed May 02 20:55:35 2018"
## [1] "designing treatments Wed May 02 20:55:35 2018"
## [1] " have initial level statistics Wed May 02 20:55:35 2018"
## [1] "design var size Wed May 02 20:55:35 2018"
## [1] "design var intake_condition Wed May 02 20:55:35 2018"
## [1] "design var outcome_condition Wed May 02 20:55:35 2018"
## [1] "design var sex_clean Wed May 02 20:55:35 2018"
## [1] "design var age_at_intake Wed May 02 20:55:35 2018"
## [1] "design var stage_at_outcome Wed May 02 20:55:35 2018"
## [1] "design var is_mix Wed May 02 20:55:35 2018"
## [1] " scoring treatments Wed May 02 20:55:35 2018"
## [1] "have treatment plan Wed May 02 20:55:35 2018"
```

```
scoreFrame_test <- treatplan_test %>%
    magrittr::use_series(scoreFrame) %>%
    select(varName, origName, code)

newvars_test <- scoreFrame_test %>%
  filter(code %in% c("clean", "lev")) %>%
  magrittr::use_series(varName)

dog_test_treat <- prepare(treatplan_test, dog_test, varRestriction = newvars_test)
```

2. Find the optimal number of trees for the xgboost model

```
library(xgboost)

cv <- xgb.cv(data = as.matrix(dog_train_treat),
             label = dog_train$placed,
             nrounds = 100,
             nfold = 5,
             objective = "reg:linear",
             eta = 0.3,
             max_depth = 6,
             early_stopping_rounds = 10,
             verbose = 0)

elog <- cv$evaluation_log

elog %>%
   summarize(ntrees.train = which.min(train_rmse_mean),
             ntrees.test  = which.min(test_rmse_mean))
```

ntrees.train <int>	ntrees.test <int>
20	10

1 row

3. Build the XGBoost model using the optimal number of trees, in this case, 10.

```
dog_xgb <- xgboost(data = as.matrix(dog_train_treat),
                   label = dog_train$placed,
                   nrounds = 10,
                   objective = "reg:linear",
                   eta = 0.3,
                   depth = 6,
                   verbose = 0)
```

4. Predict testing set outcome

```
dog_test$pred_xgb <- predict(dog_xgb,as.matrix(dog_test_treat))
```

5. Calculate the RMSE & AUC

```
rmse_xgb <- dog_test %>%
  mutate(residuals = placed - pred_xgb) %>%
  summarize(rmse = sqrt(mean(residuals^2)))

library(Metrics)
auc_xgb <- auc(actual=dog_test$placed, predicted=dog_test$pred_xgb)
print(paste0("Test set AUC XGB: ", auc_xgb))
```

```
## [1] "Test set AUC XGB: 0.866361544617847"
```

Compare the Two Models

```
## [1] "RMSE GBM: 0.28841789074812"
```

```
## [1] "RMSE XGB: 0.285649348983734"
```

```
## [1] "Test set AUC GBM: 0.873844537815126"
```

```
## [1] "Test set AUC XGB: 0.866361544617847"
```